Instructions conditionnelles et logique booléenne

Bruno Bouzy
1er septembre 2017

Exercice 1

Ecrire un programme `pair.c` qui demande un nombre entier à l'utilisateur et affiche `pair` si le nombre est pair, `impair` sinon.

Exercice 2

Quelle est la sortie du programme suivant ?

```
// operBooleen.c
#include <stdio.h>
int main() {
  int a=1, b=1, c=0, d=0;
  printf("a = %d, b = %d, c = %d, d = %d.\n", a, b, c, d);
  printf("a || b && c || d  = %d.\n", a || b && c || d);
  printf("(a||b) && c || d) = %d.\n", (a||b) && c || d);
  printf("a || (b&&c) || d  = %d.\n", a || (b&&c) || d);
  printf("a || b && (c||d)  = %d.\n", a || b && (c||d));
  return (0);
}
```

Exercice 3

1) Dans le programme `extrait.c`, que font les trois instructions `if` ?

```
// extrait.c
#include <stdio.h>
int main() {
  int a, b;
  printf("a ? "); scanf("%d", &a);
  printf("b ? "); scanf("%d", &b);
  if (a>b) a = b;          else b = a;
  if (a>b) a++;            else b++;
  if (a>b) { a++; b++; }   else a=0;
  printf("a = %d, b = %d\n", a, b);
  return (0);
}
```

2) Donner la sortie du programme `extrait.c` pour les 3 cas d'utilisation suivants:

cas n° 1:	cas n° 2:	cas n° 3:
`a ? 1`	`a ? 2`	`a ? 2`
`b ? 2`	`b ? 2`	`b ? 1`

3) En déduire ce que fait le programme `extrait.c` et le simplifier.

Exercice 4

Les deux principaux cas d'utilisation d'un programme sont :

cas n°1:

```
rentrez le nombre d articles commandes > 10
nac = 10
rentrez le prix unitaire ht > 3.5
punitaireht = 3.5
```

```
            ptotalht = 35.00
            ptotalttc = 42.00
            le prix total ttc ne depasse pas 1000
            pfinal = 42.00
```

cas n°2:

```
            rentrez le nombre d articles commandes > 100
            nac = 100
            rentrez le prix unitaire ht > 12
            punitaireht = 12.00
            ptotalht = 1200.00
            ptotalttc = 1440.00
            le prix total ttc depasse 1000
            pfinal = 1368.00
```

En identifiant les différences entre les deux cas, et en utilisant l'instruction `if`, écrire un programme `facture.c` correspondant à ces 2 cas.

Exercice 5

Ecrire un programme `ordonne3valeurs.c` qui demande 3 valeurs entières à l'utilisateur, affecte la plus grande valeur à une variable `grand`, la valeur intermédiaire à une variable `moyen` et la plus petite des trois valeurs à une variable `petit`, puis affiche les valeurs de `grand`, `moyen` et `petit`.

Exercice 6

Que fait le programme `cible.c` ?

```
// cible.c
#define CIBLE_1  1000
#define CIBLE_2  100
#include <stdio.h>
#include <math.h>
int main() {
  float x, y;
  int danslemille, dehors, total_points=0;
  printf("x ? "); scanf("%f", &x); printf("x = %.2f\n", x);
  printf("y ? "); scanf("%f", &y); printf("y = %.2f\n", y);

  float d = sqrt(x*x + y*y);
  danslemille = (d < 1);
  dehors = (d > 3);
  if (danslemille)  total_points = CIBLE_1;
  else if (!dehors) total_points = CIBLE_2;

  printf("total points = %d\n", total_points);
  return (0);
}
```

Exercice 7

Ecrire un programme `calculatrice.c` qui demande une première valeur, une opération (+ - * ou /), une deuxième valeur et affiche le résultat de l'opération sur les deux valeurs. On utilisera l'instruction `switch`.